EE 440 Homework #1

Luke Jiang (1560831)

1)

-a.

Original image:

Enlarged image:

-b.

2)

-a.

```
% 2.a. load the lena photo and display it
lena = imread('1_4.bmp');
imshow(lena);
```

-b.

```
% 2.b. get the type of lena and its maximum and minimum
disp(class(lena));
disp(max(lena(:)));
disp(min(lena(:)));
```

```
>> hw1
uint8
   255

   0
```

The class of the image is uint8

The maximum of image 1_4.bmp is 255 and the minimum is 1

-c.

```
% 2.c. convert the image to double and try to display
lena_double = double(lena);
imshow(lena_double);
```

The image displayed is almost white with a few black dots. Imshow cannot display the image properly.

-d.

```
% 2.d. normalize the doubled image to display it properly
imshow(lena_double / 255);
```

Since the imshow requires the input double to be in range [0 1] and lena_double has a maximum of 255, we must divide lean_double by 255 first before display it with imshow.

3)

-a.

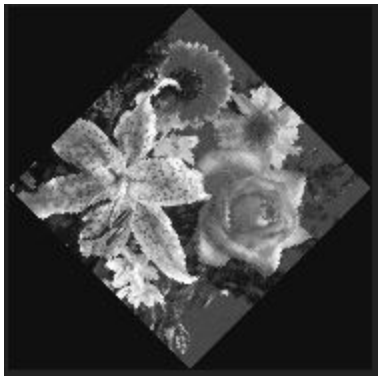
```
% 3.a. read 1_2.tif, convert to gray, save as Y
[X, map] = imread('1_2.tif', 'tif');
map_gray = rgb2gray(map);
imwrite(X, map_gray, 'Y.bmp');
```

The resultant image Y.bmp:

-b.

```
% 3.b. rotate Y by 45 degrees clockwise, save as Z
Z = imrotate(X, -45);
imwrite(Z, map_gray, 'Z.bmp');
```

The resultant image Z.bmp:



4)

-a.

```
% 4.a. reduce 1_3.asc by 4
A = load('1_3.asc');
% 4.a.i. keep one pixel out of every 4x4 pixel area
Y1 = A(1:4:384, 1:4:256);
imshow(Y1 / 256);
% 4.a.ii. replace every 4x4 pixel area by its average
Y2 = zeros(96, 64);
for i = 1:96
    for j = 1:64
        Y2(i, j) = mean2(A(i*4-3:i*4, j*4-3:j*4));
    end
end
figure; imshow(Y2 / 256);
```

The resultant images Y1 and Y2 are:

Y1 Y2

-b.

```
% 4.b. enlarge Y1 by 4
% 4.b.i. using pixel repeating
Bi = zeros(384, 256);
for i = 1:384
    for j = 1:256
        Bi(i, j) = Y1(1+floor((i-1)/4) , 1+floor((j-1)/4));
    end
end
 figure; imshow(Bi/256);

% 4.b.ii. using interpolation
% -put all values in Y1 to corresponding locations in Bii
Bii = zeros(384, 256);
for i = 1:4:384
    for j = 1:4:256
        Bii(i, j) = Y1(1+(i-1)/4, 1+(j-1)/4);
    end
end
% -interpolate along x axis
for j = 1:4:256                     % traverse all rows to be interpolated
    for i = 1:380                   % traverse each pixel in each line, except the last four
        if (mod(i-1, 4) ~= 0)       % if pixel is not from Y1, interpolate
            Q1 = floor((i-1)/4)*4+1; % get previous data point
            Q2 = Q1 + 4;            % get next data point
            Bii(i, j) = ((i-Q1)*Bii(Q2, j) + (Q2-i)*Bii(Q1, j)) / 4;
        end
    end
end
% -interpolate along y axis
for i = 1:380                       % traverse all columns
    for j = 1:252                   % traverse all pixels in each column
        if mod(j-1, 4)              % if the pixel is not from Y1, interpolate
            Q1 = floor((j-1)/4)*4+1; % get previous data point
            Q2 = Q1 + 4;            % get next data point
            Bii(i, j) = ((j-Q1)*Bii(i, Q2) + (Q2-j)*Bii(i, Q1)) / 4;
        end
    end
end
figure; imshow(Bii/256);
```

The results are:

The result generated by pixel repeating (left) looks blocky, and the result generated by interpolation (right) looks smoother. Also, the boundary of the right one is black, since these data require extra information to be interpolated.